A filter-trust-region method for simple-bound constrained optimization

Sainvitu, C.; Toint, P.L.

Published in:
Optimization Methods and Software

DOI:
10.1080/10556780701322970

Publication date:
2007

Document Version
Publisher's PDF, also known as Version of record

Optimization Methods and Software
Vol. 00, No. 00, Month 200x, 1–18

A filter-trust-region method for simple-bound constrained optimization

Caroline Sainvitu and Philippe L. Toint
(Received 00 Month 200x; In final form 00 Month 200x)

In this paper we propose a filter-trust-region algorithm for solving nonlinear optimization problems with simple bounds. It extends the technique of Gould, Sainvitu and Toint [15] designed for unconstrained optimization problems. The two main ingredients of the method are a filter-trust-region algorithm and a gradient-projection method. The algorithm is shown to be globally convergent to at least one first-order critical point. Numerical experiments on a large set of problems are also reported.

Keywords. bound-constrained optimization, filter techniques, trust-region algorithms, gradient-projection methods, convergence theory, numerical experiments

AMS subject classifications. 90C30, 65K05, 90C26, 90C06.

1 Introduction

This paper describes an algorithm which combines filter techniques, gradient-projection and trust-region methods, and which is designed for solving the following nonlinear minimization problem

$$\min_{x\in\mathbb{R}^n} \; f(x), \tag{1}$$

subject to the simple-bound constraint

$$l \leq x \leq u, \tag{2}$$

where f is a twice continuously differentiable function of the variables $x \in \mathbb{R}^n$ and l and u represent lower and upper bounds on the variables. Note that any

Department of Mathematics, University of Namur, 61, rue de Bruxelles, B-5000 Namur, Belgium (caroline.sainvitu@fundp.ac.be, philippe.toint@fundp.ac.be).

of the bounds in (2) may be infinite. Without loss of generality, we assume that $l_i < u_i$ for all $i = 1, \ldots, n$.

Filter methods have been first introduced for constrained nonlinear optimization problems by Fletcher and Leyffer [9] and they have been actually applied in many current optimization techniques [2, 8, 10, 11, 22, 23]. More recently, they have been extended by Gould, Leyffer and Toint [12, 16] to the nonlinear feasibility problem (including nonlinear least-squares and nonlinear equations) and by Gould, Sainvitu and Toint [15] to the general unconstrained optimization problem. In this paper we present and analyze a further extension of that filter-trust-region method to simple-bound constrained optimization problems. We propose combining the filter-trust-region algorithm of [15] with a gradient-projection method (see *e.g.* [3, 4, 17–19]). Note that the choice of a projection-type method is one of the possible ways to adapt this technique to bound-constrained optimization, but not the only one. We could, for example, use an interior-point method to deal with bounds.

This paper is organized as follows. In Section 2, we motivate and state the algorithm, whose global convergence to points satisfying first-order optimality conditions is shown in Section 3. Computational results are presented and discussed in Section 4. In the last section, we give some concluding remarks.

2 The algorithm

In this section we present a filter-trust-region algorithm for the solution of optimization problems subject to simple bounds. To this end, we need to define some concepts. The set of points which satisfy (2) is the *feasible box* and we denote it by $\mathcal{C}$. Any point belonging to this box is said to be *feasible*. The *"projected" gradient* of the objective function f onto the feasible box (2) is defined as

$$\bar{g}(x) \stackrel{\text{def}}{=} x - P[x - \nabla_x f(x), l, u], \tag{3}$$

where the projection operator $P[x, l, u]$ is defined componentwise by

$$P[x, l, u]_i = \begin{cases} l_i & \text{if} \quad x_i \leq l_i, \\ x_i & \text{if} \quad l_i < x_i < u_i, \\ u_i & \text{if} \quad u_i \leq x_i, \end{cases}$$

and where $\nabla_x f(x)$ denotes the gradient of the objective function. Note that the projection of any vector x onto the feasible region is extremely easy to compute when the region is a box. The projected gradient can be used to characterize first-order critical points; a point $x^* \in \mathcal{C}$ is a first-order critical

point for problem (1)-(2) if and only if

$$\bar{g}(x^*) = 0. \tag{4}$$

In what follows, we will use the following *first-order criticality measure*

$$\pi(x) \stackrel{\text{def}}{=} \|x - P[x - \nabla_x f(x), l, u]\|_\infty = \|\bar{g}(x)\|_\infty \tag{5}$$

(see *e.g.* [5, Chapter 8] and [3]).
We propose a modification of the existing filter-trust-region algorithm of Gould, Sainvitu and Toint [15], designed for unconstrained optimization, to the bound constrained case. As in this latter paper we use a multidimensional filter technique. In our context, the optimality condition (4) suggests that an iterative method for problem (1)-(2) must drive the projected gradient $\bar{g}(x_k)$ to zero for some sequence of feasible x_k. Therefore, the aim of the filter is to encourage convergence to first-order critical points by driving every component of the projected gradient

$$\bar{g}(x) = (\bar{g}_1(x), \bar{g}_2(x), \ldots, \bar{g}_n(x))^T$$

to zero.

2.1 *Computing a trial point*

Before indicating how to apply our filter technique, we start describing how to compute the trial point $x_k^+ = x_k + s_k$ from a current feasible iterate x_k. At each iteration k of the algorithm, we define the quadratic model of the objective function to be

$$m_k(x_k + s) = f(x_k) + g_k^T s + \frac{1}{2} s^T H_k s, \tag{6}$$

where g_k denotes the gradient $\nabla_x f(x_k)$ and H_k is a symmetric approximation to the Hessian matrix $\nabla_{xx} f(x_k)$. We also consider a *trust region* centered at the current iterate x_k

$$\mathcal{B}_k = \{x_k + s \mid \|s\|_\infty \leq \Delta_k\},$$

where we believe the quadratic model to be adequate. Note that we use the ℓ_∞-norm to define the trust region. A trial step s_k is then computed by finding

an approximation to the solution of the following trust-region subproblem

$$\begin{array}{ll} \text{minimize} & m_k(x_k + s) \\ \text{subject to} & l - x_k \leq s \leq u - x_k \\ & \|s\|_\infty \leq \Delta_k. \end{array} \tag{7}$$

This could be achieved by using a gradient-projection method to identify the set of active bounds, followed by a minimization of the quadratic model over the subspace of remaining free variables. The geometry of the "box" shapes of the ℓ_∞-norm and of the simple bounds may be simply exploited. We can rewrite the bounds in (7) by the following "box" constraints

$$(l_k)_i \stackrel{\text{def}}{=} \max\left(l_i - (x_k)_i, -\Delta_k\right) \leq s_i \leq \min\left(u_i - (x_k)_i, \Delta_k\right) \stackrel{\text{def}}{=} (u_k)_i \; \forall\, i = 1, \ldots, n.$$

Contrary to traditional trust-region methods, we do not require here that

$$\|s_k\|_\infty \leq \Delta_k \tag{8}$$

at every iteration of our algorithm. Some steps may not be restricted to the trust region. As it is common in trust-region methods for constrained optimization [5, Chapter 8], the convergence analysis of Section 3 requires that the step provides, at every iteration k, a *sufficient decrease on the model* of the objective function, which is to say that

$$m_k(x_k) - m_k(x_k + s_k) \geq \kappa_{\text{mdc}} \pi_k \min\left[\frac{\pi_k}{\beta_k}, \Delta_k\right], \tag{9}$$

where κ_{mdc} is a constant in $(0, 1)$, $\pi_k \stackrel{\text{def}}{=} \pi(x_k)$ and

$$\beta_k \stackrel{\text{def}}{=} 1 + \|H_k\|. \tag{10}$$

Throughout the paper, the symbol $\|\cdot\|$ denotes the ℓ_2-norm.

We are now ready to specify the computation of the trial step s_k. At each iteration, the (approximate) solution of the trust-region subproblem (7) is achieved in two stages. At the first one, the Generalized Cauchy Point (GCP) is computed in order to ensure the sufficient decrease on the model (9). This GCP is defined as the first local minimizer of the quadratic model along the *Cauchy arc* $d_k(t)$ defined as

$$d_k(t) \stackrel{\text{def}}{=} \{x \mid x = P[x_k - t g_k, l_k, u_k],\; t \geq 0\},$$

(see [3], [18] or [20]). Note that this Cauchy arc is continuous and piecewise linear. The GCP is thus computed by investigating the model behavior between successive pairs of breakpoints, that are points at which a bound is encountered along the Cauchy arc, until the model starts to rise. So further progress along the boundary of the trust region is thus only possible if the model continues to reduce. The variables which lie on their bounds at the GCP are fixed thereafter. There are efficient numerical algorithms for the GCP computation which ensure that (9) is satisfied (see [4], [17] or Section 12.2 of [5]). None of these methods requires the explicit computation or knowledge of β_k. A further reduction of the quadratic model m_k beyond that guaranteed by (9) is often desirable if fast convergence is sought. Therefore, at the second stage of the step computation, attempts are made to further reduce the quadratic model (6) by modifying the values of the remaining free variables. This may be achieved, for instance, by applying a conjugate-gradient algorithm, starting from the GCP, to the subproblem (7) with the additional restriction that the variables fixed at the GCP remain fixed throughout the process (see [1], [4], [5], [7] or [17]). To summarize, each iteration of the technique used to solve the subproblem consists of choosing a face by the gradient-projection method before exploring that face by the conjugate-gradient algorithm.

2.2 *The multidimensional filter*

Traditional trust-region algorithms evaluate the objective function at the trial point and, if the reduction achieved in the objective function is at least a fraction of that predicted by the model, the new trial point x_k^+ is accepted as the new iterate x_{k+1} and the trust-region radius Δ_k is possibly enlarged. Otherwise, if the achieved reduction is too small, the trial point is rejected and the trust-region radius is reduced. By contrast, here we prefer a filter mechanism to assess the suitability of x_k^+. Our strategy is inspired by that of [15]: we decide that a trial point x_k^+ is *acceptable for the filter* $\mathcal{F}$ if and only if

$$\forall\ \bar{g}_\ell \in \mathcal{F} \quad \exists\ j \in \{1,\ldots,n\}\ :\quad |\bar{g}_j(x_k^+)| < |\bar{g}_{\ell,j}| - \gamma_{\bar{g}}\|\bar{g}_\ell\|, \tag{11}$$

where $\gamma_{\bar{g}} \in (0, 1/\sqrt{n})$ is a small positive constant and where $\bar{g}_{\ell,j} \stackrel{\rm def}{=} \bar{g}_j(x_\ell)$. We then say that x_k^+ is not *dominated* by x_ℓ. If an iterate x_k is acceptable in the sense of (11), we may wish to add it to the *multidimensional filter*, which is a list of n-tuples of the form $(\bar{g}_{k,1},\ldots,\bar{g}_{k,n})$, such that none of the corresponding iterates is dominated by any other. We also remove from the filter every $\bar{g}_\ell \in \mathcal{F}$ such that $|\bar{g}_{\ell,j}| > |\bar{g}_{k,j}|$ for all $j \in \{1,\ldots,n\}$. We refer the reader to [15] for further details.

The mechanism described so far is adequate for convex problems because a zero projected gradient is both necessary and sufficient for second-order criticality. However, it may be unsuitable for nonconvex ones. Indeed it might prevent progress away from a saddle point, for which an increase in the projected gradient components is desirable. Therefore, as in [15], we modify the filter mechanism to ensure that the filter is reset to the empty set after each iteration giving sufficient descent on the objective function (in the sense of (9)) at which the model m_k was detected to be nonconvex, and set an upper bound on the acceptable objective function values to ensure that the obtained decrease is permanent.

2.3 *The Filter-Trust-Region Algorithm*

We are now ready to combine these ideas into an algorithm whose main objective is to let the filter play the major role in ensuring global convergence within "convex basins", and to fall back on a traditional trust-region algorithm only if things do not go well or if negative curvature is encountered.

Algorithm 1 : Filter-Trust-Region Algorithm

Step 0 : Initialization.
Let be given an initial point $x_0 \in \mathcal{C}$ and an initial trust-region radius $\Delta_0 > 0$. The constants $\gamma_{\bar{g}} \in (0, 1/\sqrt{n})$, η_1, η_2, γ_1, γ_2 and γ_3 are also given and satisfy

$$0 < \eta_1 \leq \eta_2 < 1 \quad \text{and} \quad 0 < \gamma_1 \leq \gamma_2 < 1 \leq \gamma_3. \tag{12}$$

Compute $f(x_0)$ and $\bar{g}(x_0)$, set $k = 0$. Initialize the filter $\mathcal{F}$ to the empty set and choose $f_{\sup} \geq f(x_0)$. Define two flags `RESTRICT` and `NONCONVEX`, the former to be unset.

Step 1: Determine a trial step.
Compute a finite step s_k such that $x_k + s_k \in \mathcal{C}$, that "sufficiently reduces" the model m_k, *i.e.* that satisfies (9), and that also satisfies $\|s_k\|_\infty \leq \Delta_k$ if `RESTRICT` is set or if m_k is nonconvex. In the latter case, set `NONCONVEX`; otherwise unset it. Compute the trial point $x_k^+ = x_k + s_k$.

Step 2: Compute $f(x_k^+)$ and define the following ratio

$$\rho_k = \frac{f(x_k) - f(x_k^+)}{m_k(x_k) - m_k(x_k^+)}.$$

If $f(x_k^+) > f_{\sup}$, set $x_{k+1} = x_k$, set `RESTRICT` and go to Step 4.

Step 3: Tests to accept the trial step.

- Compute $\bar{g}_k^+ = \bar{g}(x_k^+)$.
- If x_k^+ is acceptable for the filter $\mathcal{F}$ and `NONCONVEX` is unset:
  Set $x_{k+1} = x_k^+$, unset `RESTRICT` and add $\bar{g}_k^+$ to the filter $\mathcal{F}$ if either $\rho_k < \eta_1$ or $\|s_k\|_\infty > \Delta_k$.
- If x_k^+ is not acceptable for the filter $\mathcal{F}$ or `NONCONVEX` is set:
  If $\rho_k \geq \eta_1$ and $\|s_k\|_\infty \leq \Delta_k$, then

  set $x_{k+1} = x_k^+$, unset `RESTRICT` and if `NONCONVEX` is set, set $f_{\rm sup} = f(x_{k+1})$ and reinitialize the filter $\mathcal{F}$ to the empty set;

  else set $x_{k+1} = x_k$ and set `RESTRICT`.

Step 4: Update the trust-region radius.
If $\|s_k\|_\infty \leq \Delta_k$, update the trust-region radius by choosing

$$\Delta_{k+1} \in \begin{cases} [\gamma_1 \Delta_k, \gamma_2 \Delta_k] & \text{if} \quad \rho_k < \eta_1, \\ [\gamma_2 \Delta_k, \Delta_k] & \text{if} \quad \rho_k \in [\eta_1, \eta_2), \\ [\Delta_k, \gamma_3 \Delta_k] & \text{if} \quad \rho_k \geq \eta_2; \end{cases} \tag{13}$$

otherwise, set $\Delta_{k+1} = \Delta_k$. Increment k by one and go to Step 1.

As it stands, the algorithm lacks formal stopping criteria. In practice, we obviously stop the calculation if the infinity norm of the projected gradient (3) falls below some user-defined tolerance and the flag `NONCONVEX` is unset, or if some fixed maximum number of iterations is exceeded. Note that our conditions on the step in Step 1 require that we recompute the step s_k within the trust region if negative curvature is discovered for the model.

3 Global convergence to first-order critical points

We now prove that Algorithm *1* is globally convergent to at least one first-order critical point. In order to obtain our global convergence properties, we will use the following assumptions.

A1 f is twice continuously differentiable on $\mathbb{R}^n$.

A2 The iterates x_k remain in a closed, bounded domain of $\mathbb{R}^n$.

A3 For all k, the model m_k is twice differentiable on $\mathbb{R}^n$ and has a uniformly bounded Hessian.

Note that A1, A2, and A3 together imply that there exist constants $\kappa_{\rm l}$, $\kappa_{\rm u} \geq \kappa_{\rm l}$, $\kappa_{\rm ufh} \geq 1$, and $\kappa_{\rm umh} \geq 1$ such that

$$f(x_k) \in [\kappa_{\rm l}, \kappa_{\rm u}], \quad \|\nabla_{xx} f(x_k)\| \leq \kappa_{\rm ufh}, \quad \text{and} \ \ \|H_k\| \leq \kappa_{\rm umh} - 1 \tag{14}$$

for all k. Combining this with the definition of β_k, we have that

$$\beta_k \leq \kappa_{\rm umh} \tag{15}$$

for all k and all x in the convex hull of $\{x_k\}$. In what follows, we shall denote

$$\mathcal{S} = \{k \ \mid \ x_{k+1} = x_k + s_k\},$$

the set of *successful iterations*,

$$\mathcal{A} = \{k \ \mid \ \bar{g}_k^+ \ \text{is added to the filter} \ \},$$

the set of *filter iterations*,

$$\mathcal{D} = \{k \ \mid \ \rho_k \geq \eta_1\},$$

the set of *sufficient descent iterations*, and

$$\mathcal{N} = \{k \ \mid \ \texttt{NONCONVEX} \ \text{is set} \ \},$$

the set of *nonconvex iterations*. Observe that $\mathcal{A} \subseteq \mathcal{S}$, *i.e.* that $\bar{g}_k^+$ is included into the filter only at successful iterations. We also have that the mechanism of our algorithm imposes that

$$\mathcal{S} \cap \mathcal{N} = \mathcal{D} \cap \mathcal{N}. \tag{16}$$

Finally, we state a property of the algorithm which is crucial for the proofs of the next section.

LEMMA 3.1 *We have that, for all $k \geq 0$,*

$$f(x_0) - f(x_{k+1}) \geq \sum_{\substack{j=0 \\ j \in \mathcal{S} \cap \mathcal{N}}}^{k} [f(x_j) - f(x_{j+1})]. \tag{17}$$

Proof The technical proof is exactly the same as for Lemma 3.1 in [15]. □

Our convergence analysis is strongly inspired by Chapter 6 and Chapter 8 in [5] and by [15]. We devote this section to a discussion of the modification of the convergence analysis of [15] that are required to cover the bound constrained case, using the equivalence between the ℓ_2- and ℓ_∞-norms.

We begin our convergence analysis to first-order critical points by proving that, as long as a first-order critical point is not approached, we do not have infinitely many successful nonconvex iterations in the course of the algorithm. Firstly, we recall two results from [5] in order to show that the trust-region radius is bounded away from zero.

The following lemma shows that the error between the objective function and its model decreases quadratically with the trust-region radius.

LEMMA 3.2 *Suppose that* A1-A3 *hold and that* $\|s_k\|_\infty \leq \Delta_k$. *Then we have that*

$$|f(x_k + s_k) - m_k(x_k + s_k)| \leq \kappa_{\rm ubh} \Delta_k^2, \tag{18}$$

where $x_k + s_k \in \mathcal{B}_k$ *and*

$$\kappa_{\rm ubh} \stackrel{\rm def}{=} n \ \max[\ \kappa_{\rm ufh}, \kappa_{\rm umh}]. \tag{19}$$

Proof The proof is inspired by [5, Theorem 6.4.1] but, in our context, we need to make the additional assumption that $\|s_k\|_\infty \leq \Delta_k$ explicit (instead of being implicit, in this reference, in the definition of a trust-region step) and we have to use the equivalence between the ℓ_2- and ℓ_∞-norms. □

We next show that the trust-region radius must increase if the current iterate is not first-order critical and the trust-region radius is small enough.

LEMMA 3.3 *Suppose that* A1-A3 *and* (9) *hold and that* $\|s_k\|_\infty \leq \Delta_k$. *Suppose furthermore that* $\bar{g}_k \neq 0$ *and that*

$$\Delta_k \leq \frac{\kappa_{\rm mdc} \pi_k (1 - \eta_2)}{\kappa_{\rm ubh}}. \tag{20}$$

Then we have that $\rho_k \geq \eta_2$ *and*

$$\Delta_{k+1} \geq \Delta_k. \tag{21}$$

Proof The proof is the same as for Theorem 6.4.2 in [5] when $\|s_k\|_\infty \leq \Delta_k$ except that we now have to replace $\|g_k\|$ by the criticality measure π_k and that we use (9) instead of the model decrease defined in [5, Chapter 6]. The idea of the proof is to show that, as long as the current iterate is not a first-order

critical point, and that the radius satisfies (20), the iteration must be very successful, and the trust-region radius is enlarged according to (13). □

Consequently, we may now obtain that the trust-region radius cannot become arbitrarily small if the iterates stay away from first-order critical points.

LEMMA 3.4 *Suppose that* A1-A3 *and* (9) *hold and that there exists a constant* $\kappa_{\rm lbg} > 0$ *such that* $\pi_k \geq \kappa_{\rm lbg}$ *for all* k. *Then there is a constant* $\kappa_{\rm lbd} > 0$ *such that*

$$\Delta_k \geq \kappa_{\rm lbd} \tag{22}$$

for all k.

Proof The proof is by contradiction and uses Lemma 3.3. It is identical to that of Lemma 3.4 in [15] except that we use the ℓ_∞-norm of the step instead of the ℓ_2-norm and that we now have to replace $\|g_k\|$ by the criticality measure π_k. □

We now prove the essential result that the number of successful nonconvex iterations must be finite unless a first-order critical point is approached.

THEOREM 3.5 *Suppose that* A1-A3 *and* (9) *hold and that there exists a constant* $\kappa_{\rm lbg} > 0$ *such that* $\pi_k \geq \kappa_{\rm lbg}$ *for all* k. *Then there can only be finitely many successful nonconvex iterations in the course of the algorithm, i.e.* $|\mathcal{S} \cap \mathcal{N}| < +\infty$.

Proof The proof is inspired by [15, Theorem 3.5] except that $\|g_k\|$ is replaced by π_k and we now use condition (9) on the model reduction. □

We now establish the criticality of the limit point of the sequence of iterates when there are only finitely many successful iterations.

THEOREM 3.6 *Suppose that* A1-A3 *and (9) hold and that there are only finitely many successful iterations, i.e.* $|\mathcal{S}| < +\infty$. *Then* $x_k = x^*$ *for all sufficiently large* k, *and* x^* *is first-order critical.*

Proof The proof is identical to that of Theorem 3.6 in [15] except that we have to replace $\|g_k\|$ by the criticality measure π_k and that we use the ℓ_∞-norm of the step instead of the ℓ_2-norm. □

Having proved the desired convergence property for the case where $\mathcal{S}$ is finite, we restrict our attention, for the rest of this section, to the case where the filter is updated an infinite number of times, *i.e.* $|\mathcal{S}| = +\infty$. We start by investigating what happens if infinitely many values are added to the filter in the course of the algorithm, *i.e.* $|\mathcal{A}| = +\infty$.

THEOREM 3.7 *Suppose that* A1-A3 *and (9) hold and that* $|\mathcal{A}| = |\mathcal{S}| = +\infty$. *Then*

$$\liminf_{k\to\infty} \pi_k = 0. \tag{23}$$

Proof The proof is the same as for Theorem 3.7 in [15] except that $\|g_k\|$ is replaced by π_k and that we use the new filter acceptance definition (11). The proof is by contradiction. We suppose that, for all k large enough, $\pi_k \geq \kappa_{\text{lbg}}$ for some $\kappa_{\text{lbg}} > 0$. Theorem 3.5 implies that the filter is no longer reset to the empty set for k sufficiently large. By using the filter test acceptance mechanism and our initial assumption, we can derive a contradiction exactly as in [15, Theorem 3.7]. □

Consider now the case where the number of iterates added to the filter in the course of the algorithm is finite.

THEOREM 3.8 *Suppose that* A1-A3 *and (9) hold and that* $|\mathcal{S}| = +\infty$ *but* $|\mathcal{A}| < +\infty$. *Then (23) holds.*

Proof Again the proof is identical to that of Theorem 3.8 in [15]. □

The preceding two results show that at least one of the limit points of the sequence of iterates generated by the algorithm satisfies the first-order necessary condition. However this result cannot be improved to obtain that all limit points are first-order critical without affecting the algorithm's numerical behavior (see the example in [15]).

4 Numerical experiments

In this section, we report the computational results obtained by running our algorithm on a set of 109 simple-bound constrained problems[1] from the CUTEr collection [13]. These problems are either academic test cases or arise from applications. Their names and dimensions are given in Table 1. Since a variable can be fixed (*i.e.* its upper and lower bounds are equal, and therefore the variable is not allowed to change), we consider the dimension of a problem as the number of variables minus the number of fixed ones. The dimension of the problems varies from 1 to 11130.

We always use the starting point supplied with the problem. However, if this initial point is not feasible, we project it onto the feasible box. All tests were performed in double precision on a workstation with a 3.2 GHz Pentium IV

[1] Two problems, namely `CHARDIS0` and `HARKERP2`, were removed because they could not be run within the memory limits of the testing machine by any of the codes.

Table 1. The test problems and their dimension.

Problem	n	Problem	n	Problem	n
3PK	30	JNLBRNGB	9604	PALMER5B	9
ALLINIT	3	LINVERSE	1999	PALMER5D	8
BDEXP	5000	LOGROS	2	PALMER5E	8
BIGGSB1	5000	MAXLIKA	8	PALMER6A	6
BQP1VAR	1	MCCORMCK	5000	PALMER6E	8
BQPGABIM	46	MDHOLE	2	PALMER7A	6
BQPGASIM	50	MINSURFO	5002	PALMER7E	8
BQPGAUSS	2003	NCVXBQP1	10000	PALMER8A	6
CAMEL6	2	NCVXBQP2	10000	PALMER8E	8
CHEBYQAD	100	NCVXBQP3	10000	PENTDI	5000
CHENHARK	5000	NOBNDTOR	5184	PROBPENL	500
CVXBQP1	10000	NONSCOMP	5000	PSPDOC	4
DECONVB	61	OBSTCLAE	9604	QR3DLS	610
EG1	3	OBSTCLAL	9604	QRTQUAD	5000
EXPLIN	1200	OBSTCLBL	9604	QUDLIN	5000
EXPLIN2	1200	OBSTCLBM	9604	S368	8
EXPQUAD	1200	OBSTCLBU	9604	SCOND1LS	5000
HADAMALS	380	OSLBQP	8	SIM2BQP	1
HART6	6	ODNAMUR	11130	SIMBQP	2
HATFLDA	4	PALMER1	4	SINEALI	1000
HATFLDB	4	PALMER1A	6	SPECAN	9
HATFLDC	25	PALMER1B	4	TORSION1	5184
HIMMELP1	2	PALMER1E	8	TORSION2	5184
HS1	2	PALMER2	4	TORSION3	5184
HS110	200	PALMER2A	6	TORSION4	5184
HS2	2	PALMER2B	4	TORSION5	5184
HS25	3	PALMER2E	8	TORSION6	5184
HS3	2	PALMER3	4	TORSIONA	5184
HS38	4	PALMER3A	6	TORSIONB	5184
HS3MOD	2	PALMER3B	4	TORSIONC	5184
HS4	2	PALMER3E	8	TORSIOND	5184
HS45	5	PALMER4	4	TORSIONE	5184
HS5	2	PALMER4A	6	TORSIONF	5184
JNLBRNG1	9604	PALMER4B	4	WEEDS	3
JNLBRNG2	9604	PALMER4E	8	YFIT	3
JNLBRNGA	9604	PALMER5A	8		

biprocessor and 2 Gbytes of memory under Suse Professional 9.0 Linux and the Lahey Fortran compiler (version L6.10a) with default options. We have limited all attempts to solve the test problems to a maximum of 1000 iterations or 1 hour of CPU time. The variability of CPU times for small times is taken into account by repeatedly solving the same problem until a threshold of ten seconds is exceeded and then taking the average time per run.
The values for the constants of Algorithm *1* used in our tests are

$$\gamma_1 = 0.0625,\ \gamma_2 = 0.25,\ \gamma_3 = 2,\ \eta_1 = 0.01,\ \eta_2 = 0.9,\ \Delta_0 = 1$$

and

$$\gamma_{\bar{g}} = \min\left[0.001, \frac{1}{2\sqrt{n}}\right].$$

We also choose

$$f_{\sup} = \min(10^6|f(x_0)|, f(x_0) + 1000)$$

at Step 0 of the algorithm.

We have tested two particular variants. The first one, named *filter*, is the algorithm as described in Section 2, where exact first and second derivatives are used. As we have already mentioned, at each iteration, the trial point is computed by approximately minimizing the subproblem (7). This computation is accomplished in a two-stage approach: first, we use the gradient-projection method to identify variables that will be fixed at their bounds; then the quadratic model of the objective function is further reduced with respect to the free variables by using a conjugate-gradient algorithm (see [3]). This iterative method is terminated at the first s for which

$$\|(\nabla m_k(x_k + s))_{\text{ free}}\|_\infty \leq \min\left[0.1, \max(\sqrt{\varepsilon_M}, \|\ \bar{g}(x_k)\|_\infty)\right] \|\bar{g}(x_k)\|_\infty, \tag{24}$$

where $(\nabla m_k(x_k+s))_{\text{ free}}$ denotes the restricted gradient of the quadratic model with respect to the remaining free variables[1] at the beginning of the conjugate-gradient iteration and ε_M is the machine precision. Based on practical experience [16], we also impose that $\|s_k\|_\infty \leq 1000\ \Delta_k$ at all iterations following the first one at which a restricted step is taken. Every run of the algorithm was terminated if the infinity norm of the projected gradient falls below some tolerance, *i.e.* if

$$\|\bar{g}(x_k)\|_\infty \leq 10^{-6}. \tag{25}$$

Finally, dominated filter points are always removed from the filter in our tests.

The second algorithmic variant is the *pure trust-region* one, that is the same algorithm with the exception that trial points are never acceptable for the filter and the flag `RESTRICT` is always set, which is to say that steps are always restricted within the trust region. This variant is therefore analogous to a classical trust-region method.

On the 107 problems, the filter variant successfully solves 101 problems and the pure trust-region one 100. In order to produce the performance profiles given in this section, we have excluded four problems from the 107 box-constrained problems. We have first removed `HS25` from the test set because the starting point supplied with the problem is already a stationary point.

[1]The remaining free variables are those which are not fixed at the GCP.

Three other problems have also been taken away because both variants did not report the same final objective function value for them. These problems are MAXLIKA, PALMER3 and PALMER4. Note that the last two are both least-squares problems, so the fact that filter and pure trust-region variants stop at different solutions is probably due to the ill-conditioning of these problems. Having excluded the above-mentioned problems, both variants report the same final objective function value for problems where they both succeed. Both variants fail on BIGGSB1, PALMER5A, PALMER7A, QRTQUAD and SCOND1LS because the maximum number of iterations has been reached before convergence is declared. The filter variant also fails, for the same reason, on MINSURFO, and the pure trust-region algorithm stalls on PALMER5B and PALMER5E. Furthermore, the pure trust-region variant is also unable, for problems EXPLIN, EXPQUAD and PALEMR1A, to reduce the infinity norm of the projected gradient sufficiently to meet the stopping criterion (25) even though the objective function value obtained is very close to the problem solution. However we have counted these occurrences as successful in the discussion of this section.

Figures 1, 2 and 3 give the performance profiles in term of number of iterations, CPU time and the total amount of conjugate-gradient iterations, respectively. Performance profiles give, for every $\sigma \geq 1$, the proportion $p(\sigma)$ of test problems on which each considered algorithmic variant has a performance within a factor σ of the best (see [6] for a more complete discussion).

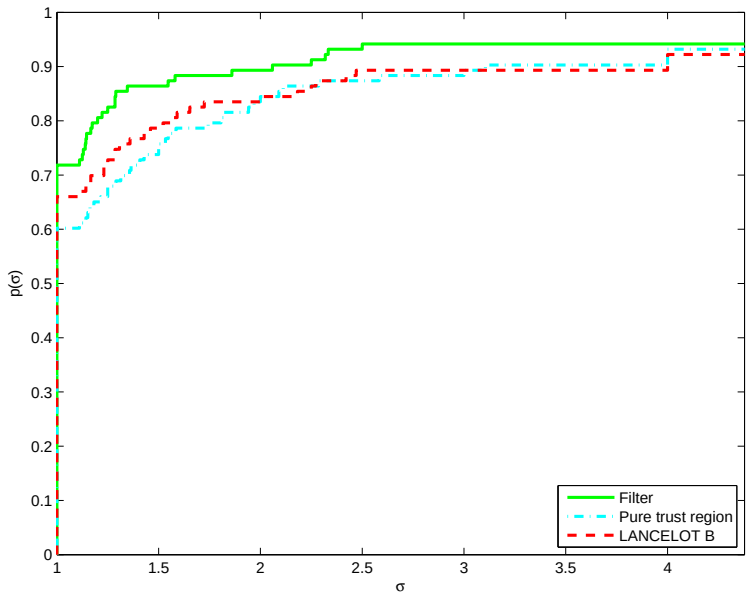

Figure 1. Iterations performance profile for both variants and LANCELOT-B.

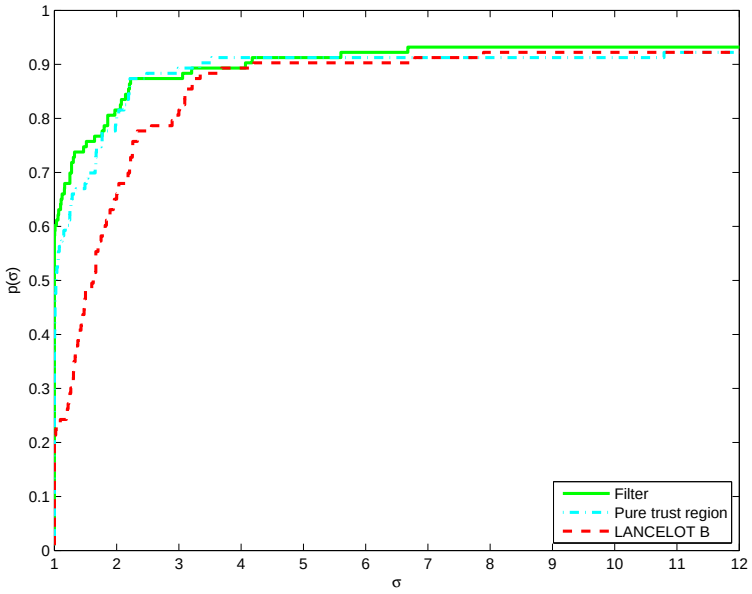

Figure 2. CPU performance profile for both variants and LANCELOT-B.

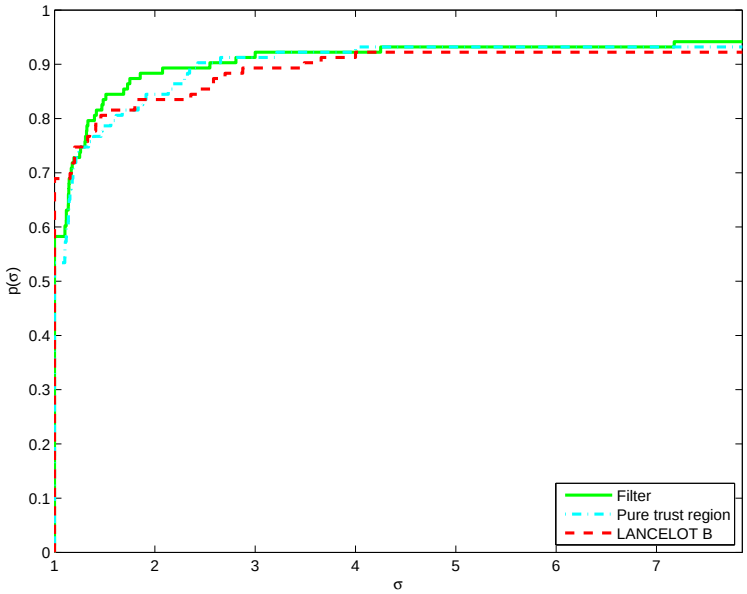

Figure 3. CG iterations performance profile for both variants and LANCELOT-B.

Although the numerical results are not as significant as for the algorithm in the unconstrained case (see the performance profiles in [15]), we obtain interesting results. We can see on these figures that the filter variant is significantly more efficient than the pure trust-region method in term of the number of iterations (which is identical to the number of function evaluations minus one). Its advantage is smaller in term of conjugate-gradient iterations and CPU-time efficiency.
We also remark from our numerical tests that the maximum number of filter entries does not exceed 5 for 76 problems, lies between 6 and 10 for 12 problems, between 11 and 30 for 11 problems and exceeds 30 for only two problems: `EXPQUAD` (31 entries) and `PALMER5E` (50 entries). Note that the pure trust-region variant does not solve the last one. Moreover, we did not observe any obvious correlation between filter size and number of variables. It should also be observed that, for the majority of problems where the filter variant fails, the algorithm puts a large number of entries in the filter. However, for those problems[1], the pure trust-region variant also fails.

We also include a comparison with LANCELOT-B, one of the GALAHAD codes [14]. This is a non-monotone trust-region algorithm (see [21] or [5, Section 10.1]), which we used unpreconditioned with $\Delta_0 = 1$ and with its other settings at their default values. In order to be comparable with our code for which the stopping criterion is given in (25), the accuracy on the projected gradient in LANCELOT-B is set to 10^{-6}. This method, which successfully solves 99 out of 107 problems, appears to be slightly inferior to the new filter algorithm in term of number of iterations and especially in term of CPU-time efficiency. Nevertheless, LANCELOT-B is more efficient in term of conjugate-gradient iterations. As the filter variant, LANCELOT-B does not solve `BIGGSB1`, `PALMER5A`, `PALMER7A`, `QRTQUAD` or `SCOND1LS` either. It also fails on `CHENHARK`, `PALMER5B` and `PALMER5E`.

5 Conclusion

In this paper, we have proposed an algorithm for the minimization of simple-bound constrained optimization problems. The underlying idea of our algorithm is to combine three tools of nonlinear programming, namely trust regions, gradient-projection methods and filter techniques. We have shown that, under standard assumptions, it produces at least a first-order critical point, irrespective of the chosen starting point. A second-order convergence analysis remains to be done but difficulties are expected since one knows that possibly only one limit point is first-order critical. The preliminary numerical results

[1] Except for `MINSURFO`.

obtained on the set of bound-constrained test problems are reported and discussed, showing a general good performance of the algorithm.

Acknowledgement

The authors wish to thank Nick Gould and Oleg Burdakov for a number of useful comments and suggestions.

References

[1] M. Andretta, E. G. Birgin, and J. M. Martnez. Practical active-set euclidian trust-region method with spectral projected gradients for bound-constrained optimization. *Optimization*, 54(3):305–325, 2005.

[2] Ch. M. Chin and R. Fletcher. Convergence properties of SLP-filter algorithms that takes EQP steps. *Mathematical Programming, Series A*, 96(1):161–177, 2003.

[3] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. Global convergence of a class of trust region algorithms for optimization with simple bounds. *SIAM Journal on Numerical Analysis*, 25(182):433–460, 1988. See also same journal 26:764–767, 1989.

[4] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. Testing a class of methods for solving minimization problems with simple bounds on the variables. *Mathematics of Computation*, 50:399–430, 1988.

[5] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *Trust-Region Methods*. Number 01 in MPS-SIAM Series on Optimization. SIAM, Philadelphia, USA, 2000.

[6] E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2):201–213, 2002.

[7] Z. Dostál. Box constrained quadratic programming with proportioning and projections. *SIAM Journal on Optimization*, 7(3):871–887, 1997.

[8] R. Fletcher, N. I. M. Gould, S. Leyffer, Ph. L. Toint, and A. Wächter. Global convergence of trust-region SQP-filter algorithms for nonlinear programming. *SIAM Journal on Optimization*, 13(3):635–659, 2002.

[9] R. Fletcher and S. Leyffer. Nonlinear programming without a penalty function. *Mathematical Programming*, 91(2):239–269, 2002.

[10] R. Fletcher, S. Leyffer, and Ph. L. Toint. On the global convergence of a filter-SQP algorithm. *SIAM Journal on Optimization*, 13(1):44–59, 2002.

[11] C. C. Gonzaga, E. Karas, and M. Vanti. A globally convergent filter method for nonlinear programming. *SIAM Journal on Optimization*, 14(3):646–669, 2003.

[12] N. I. M. Gould, S. Leyffer, and Ph. L. Toint. A multidimensional filter algorithm for nonlinear equations and nonlinear least-squares. *SIAM Journal on Optimization*, 15(1):17–38, 2005.

[13] N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEr, a constrained and unconstrained testing environment, revisited. *ACM Transactions on Mathematical Software*, 29(4):373–394, 2003.

[14] N. I. M. Gould, D. Orban, and Ph. L. Toint. GALAHAD—a library of thread-safe Fortran 90 packages for large-scale nonlinear optimization. *ACM Transactions on Mathematical Software*, 29(4):353–372, 2003.

[15] N. I. M. Gould, C. Sainvitu, and Ph. L. Toint. A filter-trust-region method for unconstrained optimization. *SIAM Journal on Optimization*, 16(2):341–357, 2005.

[16] N. I. M. Gould and Ph. L. Toint. FILTRANE, a Fortran 95 filter-trust-region package for solving nonlinear least-squares problems and nonlinear feasibility problems. *ACM Transactions on Mathematical Software*, 2007.

[17] C. Lin and J. J. Moré. Newton's method for large bound-constrained optimization problems. *SIAM Journal on Optimization*, 9(4):1100–1127, 1999.

[18] J. J. Moré. Trust regions and projected gradients. In M. Iri and K. Yajima, editors, *System Modelling and Optimization*, volume 113, pages 1–13, Heidelberg, Berlin, New York, 1988. Springer Verlag. Lecture Notes in Control and Information Sciences.

[19] J. J. Moré and G. Toraldo. On the solution of large quadratic programming problems with bound constraints. *SIAM Journal on Optimization*, 1(1):93–113, 1991.

[20] Ph. L. Toint. Global convergence of a class of trust region methods for nonconvex minimization in Hilbert space. *IMA Journal of Numerical Analysis*, 8(2):231–252, 1988.

[21] Ph. L. Toint. A non-monotone trust-region algorithm for nonlinear optimization subject to convex constraints. *Mathematical Programming*, 77(1):69–94, 1997.

[22] M. Ulbrich, S. Ulbrich, and L. N. Vicente. A globally convergent primal-dual interior point filter method for nonconvex nonlinear programming. *Mathematical Programming, Series B*, 100(2):379–410, 2004.

[23] A. Wächter and L. T. Biegler. Global and local convergence of line search filter methods for nonlinear programming. Technical Report CAPD B-01-09, Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, USA, 2001. Available on http://www.optimization-online.org/DB_HTML/2001/08/367.html.